TÓM LƯỢC BÀI GIẢNG

(Vũ Quốc Hoàng)

CON TRỎ HÀM

Chủ đề

- Con trỏ hàm

Tài liệu

[1] Paul Deiteil, Harvey Deiteil, *C how to program*, Pearson, 8^{th} global edition, 2016.

[2] Vũ Quốc Hoàng, *Bí kíp luyện Lập trình C (Quyển 1)*, hBook, 2017.

Đọc tài liệu

- Con trỏ hàm: Mục 7.12 [1]

Kiến thức

- Đoạn mã minh họa 1

```
#include <iostream>
using namespace std;

typedef bool (CheckFunc)(int);

void select(int a[], int &n, CheckFunc *check)
{
    int cur = 0;
    for(int i = 0; i < n; i++)
        if(check(a[i]))
            a[cur++] = a[i];
    n = cur;
}

void output(int a[], int n)
{
    for(int i = 0; i < n; i++)
        cout << a[i] << " ";
}

bool positive(int x) { return (x > 0); }
bool even(int x) { return (x % 2 == 0); }

int main()
{
    int a[100];
```

```
    int n;

    cout << "Nhap n: ";
    cin >> n;
    for(int i = 0; i < n; i++)
        cin >> a[i];
    cout << "\nCac so da nhap: "; output(a, n);

    select(a, n, &positive);
    cout << "\nCac so duong: "; output(a, n);

    select(a, n, even);
    cout << "\nCac so chan: "; output(a, n);
}
```

- *Kiến trúc máy Von Neumann* (Von Neumann architecture) cho phép xem chương trình như là dữ liệu: các *chỉ thị máy* (instruction) cũng được lưu trên *bộ nhớ chính* (main memory) như là *dữ liệu* (data). Hàm là dãy các chỉ thị được lưu trên bộ nhớ, do đó ta cũng có thể truy cập đến các hàm bằng con trỏ đến vùng nhớ chứa chúng, gọi là *con trỏ hàm* (function pointer).
- Tương tự như *kiểu dữ liệu* (data type) xác định dạng dữ liệu, *kiểu hàm* (function type) xác định dạng hàm, tức là số lượng tham số, kiểu của từng tham số và kiểu trả về của hàm. Chẳng hạn, trong ví dụ trên, kiểu hàm `CheckFunc` xác định "dạng hàm kiểm tra trên số nguyên": nhận một số nguyên và trả về giá trị luận lý (`bool`) cho biết nó có thỏa mãn tính chất nào đó không. Các hàm `positive` và `even` ở trên là các hàm có kiểu như vậy. Hàm `positive` kiểm tra tính chất "dương" và hàm `even` kiểm tra tính chất "chẵn".
- Tương tự như dữ liệu có thể được truy cập trực tiếp bằng tên biến hoặc gián tiếp bằng con trỏ dữ liệu (con trỏ thông thường đã học), hàm có thể được gọi thực thi trực tiếp bằng tên hàm hoặc gián tiếp bằng con trỏ hàm. Ở trên thì hàm `select` có tham số `check` là con trỏ đến hàm có kiểu `CheckFunc`. Ta gọi hàm mà `check` trỏ đến với đối số `a[i]` bằng toán tử giải tham chiếu (`*`) là `(*check)(a[i])` hoặc gọn hơn là `check(a[i])`. Khi gọi hàm `select` ta cần đưa địa chỉ hàm kiểm tra cụ thể cho tham số `check` bằng toán tử lấy địa chỉ (`&`) như `select(a, n, &positive)` hoặc gọn hơn, chỉ dùng tên hàm `select(a, n, positive)` như cách dùng tên mảng. (Tên hàm có thể xem là hằng con trỏ hàm tương tự như tên mảng có thể xem là hằng con trỏ dữ liệu.)
- Đoạn mã minh họa 2

```
#include <iostream>
#include <vector>
#include <string>
using namespace std;

typedef void (CommandFunc)(int&);

struct MenuItem
```

```
{
    string name;
    char key;
    CommandFunc *command;
};

void command1(int &x) { x++; }
void command2(int &x) { x *= 2; }
void command3(int &x) { x = -x; }

bool loop(vector<MenuItem> &items, int &status)
{
    cout << "Status: " << status << endl;

    for(int i = 0; i < items.size(); i++)
        cout << "Key " << items[i].key << " => " << items[i].name << ". ";
    cout << "\nEnter key: ";
    char k;
    cin >> k;

    for(int i = 0; i < items.size(); i++)
        if(k == items[i].key)
        {
            items[i].command(status);
            return true;
        }

    return false;
}

int main()
{
    vector<MenuItem> items;

    items.push_back({"Increment", '1', command1});
    items.push_back({"Double", '2', command2});
    items.push_back({"Negation", '3', command3});

    int status = 0;
    while(loop(items, status));

    cout << "Bye!";
}
```

- Chương trình trên minh họa một hệ thống "quản lý menu" dựa trên con trỏ hàm.
- Kiểu hàm và con trỏ hàm là các đặc trưng rất mạnh, làm nền tảng cho các kĩ thuật phổ biến (sẽ học sau này) như *hàm gọi lại* (callback function), *vòng lặp thông điệp* (message loop), *bộ quản lý sự kiện* (event handler), *kĩ thuật lập trình hướng sự kiện* (event-driven programming), *kĩ thuật lập trình hướng đối tượng* (object-oriented programming), …

Kĩ năng

- Thành thạo việc dùng con trỏ hàm
- Biết cách vận dụng con trỏ hàm để tổ chức chương trình trong một số tình huống

Lưu ý

- Kiểu hàm và con trỏ hàm là các đặc trưng rất mạnh, làm nền tảng cho các kĩ thuật phổ biến sẽ học sau này nên sinh viên cần rèn luyện nhiều để quen thuộc và thành thạo

Bài tập

Viết lại chương trình "Bầu Cua" trong phần mở rộng Bài 2.6 [2] theo cách hay hơn!